Optimal Testing/Maintenance Design in a Software Development Project (Development of the Dynamic Systems under Uncertainty)

AUTHOR(S):

Rinsaka, Koichiro; Dohi, Tadashi

数理解析研究所講究録 1383 巻 2004 年 72-80

Optimal Testing/Maintenance Design in a Software Development Project

林坂弘一郎, 土肥 正
Koichiro Rinsaka and Tadashi Dohi
Department of Information Engineering, Graduate School of Engineering,
Hiroshima University, Japan

1 Introduction

It is important to determine the optimal time when software testing should be stopped and when the system should be delivered to a user or a market. This problem, called *optimal software release problem*, plays a central role for the success or failure of a software development project. Okumoto and Goel [1] assumed that the number of software faults detected in the testing phase is described by an exponential software reliability model based on a non-homogeneous Poisson process (NHPP) [2], and derived an optimal software release time which minimizes the total expected software cost. Koch and Kubat [3] considered the similar problem for the other software reliability model by Jelinski and Moranda [4]. Bai and Yun [5] calculated the optimal number of faults detected before the release under the Jelinski and Moranda model. Many authors formulated the optimal software release problems based on different model assumptions and/or several software reliability models [6, 7].

It is difficult to detect and remove all faults remaining in a software during the testing phase, because exhaustive testing of all executable paths in a general program is impossible. Once the software is released to users, however the software failures may occur even in the operational phase. It is common for software developers to provide maintenance service during the period when they are still responsible for fixing software faults causing failures. In order to carry out the maintenance in the operational phase, the software developer has to keep a software maintenance team. At the same time, the management cost in the operational phase should be reduced as much as possible, but the human resources should be utilized effectively. Although the problem which determines the maintenance period is important from the practical point of view, only a very few authors paid their attention to this problem.

Kimura *et al.* [8] considered the optimal software release problem in the case where the software warranty period is a random variable. Pham and Zhang [9] developed a software cost model with both warranty and risk. They focused on the problem for determining when to stop the software testing under a warranty contract. However, it is noted that the software developer has to design the warranty contract itself and often provides the posterior service for users after software failures. Dohi *et al.* [10] formulated the problem for determining the optimal software warranty period which minimizes the total expected software cost under the assumption that the debugging process in the testing phase is described by an NHPP. Since the user's operational environment is not always same as that assumed in the software development phase, however, the above literature did not take account of the difference between two different phases.

Several reliability assessment methods during the operational phase have been proposed by some authors [11, 12]. In this paper, we develop a stochastic model for designing the optimal testing and maintenance periods, where the difference between the software testing environment and the operational environment are reflected. Based on the idea in Okamura *et al.* [12], we formulate the total expected software cost incurred for the software developer, and derive analytically the optimal testing period (release time) which minimizes the total expected software cost. We call the time length to complete the operational maintenance after the release *a planned maintenance limit*, and also derive the optimal planned maintenance limit which minimizes the total expected software cost. Throughout numerical examples, we calculate numerically the joint optimal policy combined by testing period and planned maintenance limit. Finally, the paper is concluded with some remarks.

2 Model Description

First, we make the following assumptions on the software fault-detection process:

(a) In each time when a software failure occurs, the software fault causing the failure is detected and removed immediately.

(b) The number of initial faults contained in the software program, N_0, is given by the Poisson distributed random variable with mean ω (> 0).

(c) The time to detect each software fault is independent and identically distributed nonnegative random variable with (absolutely continuous) probability distribution function $F(t)$ and density function $f(t)$.

Let $\{N(t), t \geq 0\}$ be the cumulative number of software faults detected up to time t. From the above assumptions, the probability math function of $N(t)$ is given by

$$\Pr\{N(t) = m\} = \frac{[\omega F(t)]^m e^{-\omega F(t)}}{m!} \quad m = 0, 1, 2, \cdots. \tag{1}$$

Hence, the stochastic process $\{N(t), t \geq 0\}$ is equivalent to an NHPP with mean value function $\omega F(t)$, where the fault-detection rate (debugging rate) per unit of time is given by

$$r(t) = \frac{f(t)}{1 - F(t)}. \tag{2}$$

Suppose that a software testing is started at time 0 and terminated at time t_0 (≥ 0). The time length of software life cycle t_L (> 0) is known in advance and is assumed to be sufficiently larger than t_0. More precisely, the software life cycle is measured from the time t_0. The software developer is responsible to the maintenance service for all the software failures that may occur during the software life cycle under a maintenance contract. We suppose that the project manager decides to break up the maintenance team at time $t_0 + t_W$ ($0 \leq t_W \leq t_L$) for reduction of the operational cost to keep it, but a large amount of debugging cost after the planned maintenance limit if the software failure occurs may be needed. Further, we define the following cost components;

c_0 (> 0): cost to remove each fault in the testing phase,

c_W (> 0): cost to remove each fault before the planned maintenance limit,

c_L (> 0): cost to remove each fault after the planned maintenance limit,

k_0 (> 0): testing cost per unit of time,

k_W (> 0): operational cost to keep the maintenance team per unit of time.

In the following section, we formulate the total expected software cost by introducing the above cost factors in testing and operational phases. We derive the optimal software testing period or the optimal planned maintenance limit which minimizes the total expected software cost at the end of the software life cycle. Then, we calculate the joint optimal policy combined by both testing period and planned maintenance limit.

3 Total Expected Software Cost

We formulate the total expected software cost which can occur in both testing and operational phases. In the operational phase, we consider two cost factors; the maintenance cost due to the software failure and the operational cost to keep the maintenance team.

From Eq.(1), the probability math function of the number of software faults detected during the testing phase is given by

$$\Pr\{N(t_0) = m\} = \frac{[\omega F(t_0)]^m e^{-\omega F(t_0)}}{m!}. \tag{3}$$

It should be noted that the operational environment after the release may differ from the debugging environment in the testing phase. This difference is similar to that between the accelerated life testing environment and the normal operating environment for hardware products. We introduce the environment factor a (> 0) which express the relative severity in the operational environment after the release,

and consider that the times in the testing phase and the operational phase have a proportional relationship. Namely, the time length t in the operational phase corresponds to at in the testing phase. Under the above assumption, $a = 1$ means the equivalence between the testing and operational environments. On the other hand, $a > 1$ ($a < 1$) implies that the operational environment is severe (looser) than the testing environment. Okamura *et al.* (2001) apply this technique to model the operational phase of the software, and estimate the software reliability through an example in the actual software development project. The probability math function of the number of software faults detected before the planned maintenance limit is given by

$$\Pr\{N(t_0 + t_W) - N(t_0) = m\} = \frac{\{\omega\left[F(t_0 + at_W) - F(t_0)\right]\}^m}{m!} e^{-\omega[F(t_0+at_W)-F(t_0)]}. \tag{4}$$

Similarly, the fault-detection process of the software after the planned maintenance limit is expressed by

$$\begin{aligned} &\Pr\{N(t_0 + t_L) - N(t_0 + t_W) = m\} \\ &= \frac{\{\omega\left[F(t_0 + at_L) - F(t_0 + at_W)\right]\}^m}{m!} e^{-\omega[F(t_0+at_L)-F(t_0+at_W)]}. \end{aligned} \tag{5}$$

From Eqs.(3), (4) and (5), the total expected software cost is given by

$$\begin{aligned} C(t_0, t_W) &= k_0 t_0 + c_0\omega F(t_0) + k_W t_W + c_W\omega\left[F(t_0 + at_W) - F(t_0)\right] \\ &\quad + c_L\omega\left[F(t_0 + at_L) - F(t_0 + at_W)\right]. \end{aligned} \tag{6}$$

4 Determination of the Optimal Policies

In this section we derive the optimal testing period or the optimal planned maintenance limit which minimizes the total expected software cost incurred to the software developer at the end of software life cycle. Suppose that the time to detect each software fault obeys the exponential distribution [2] with mean $1/\lambda$ (> 0). In this case, the total expected software cost in Eq.(6) becomes

$$\begin{aligned} C(t_0, t_W) &= k_0 t_0 + c_0\omega\left(1 - e^{-\lambda t_0}\right) + k_W t_W \\ &\quad + c_W\omega e^{-\lambda t_0}\left(1 - e^{-\lambda a t_W}\right) + c_L\omega e^{-\lambda t_0}\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right). \end{aligned} \tag{7}$$

We make the following assumptions:

(A-I) $c_L > c_W > c_0$,

(A-II) $c_W\left(1 - e^{-\lambda a t_L}\right) > c_0$,

(A-III) $c_W\left(1 - e^{-\lambda a t_W}\right) + c_L\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right) > c_0$.

Define

$$Q(t_W) = k_0 + \omega\lambda\left[c_0 - c_W\left(1 - e^{-\lambda a t_W}\right) - c_L\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right)\right]. \tag{8}$$

Then the following result provides the optimal software testing policy which minimizes the total expected software cost.

Theorem 1: *When the software fault-detection time distribution follows the exponential distribution with mean $1/\lambda$, under the assumptions (A-I) to (A-III), the optimal software testing period (release time) which minimizes the total expected software cost is given as follows:*

(1) If $Q(t_W) < 0$, then there exists a finite and unique optimal software testing period (release time) t_0^* (> 0), and its associated expected cost is given by

$$\begin{aligned} C(t_0^*, t_W) &= k_0 t_0^* + c_0\omega\left(1 - e^{-\lambda t_0^*}\right) + k_W t_W \\ &\quad + c_W\omega e^{-\lambda t_0^*}\left(1 - e^{-\lambda a t_W}\right) + c_L\omega e^{-\lambda t_0^*}\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right). \end{aligned} \tag{9}$$

(2) If $Q(t_W) \geq 0$, then the optimal policy is $t_0^* = 0$ with

$$C(t_0^*, t_W) = k_W t_W + c_W\omega\left(1 - e^{-\lambda a t_W}\right) + c_L\omega\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right). \tag{10}$$

Proof: For the total expected cost in Eq.(7), we have

$$C(0,t_W) = k_W t_W + c_W\omega\left(1-e^{-\lambda t_W}\right) + c_L\omega\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right) \tag{11}$$

and

$$\lim_{t_0\to+\infty} C(t_0,t_W) = +\infty. \tag{12}$$

Differentiating $C(t_0,t_W)$ with respect to t_0 yields

$$\begin{aligned}\frac{\partial C(t_0,t_W)}{\partial t_0} &= k_0 + c_0\omega\lambda e^{-\lambda t_0} - c_W\omega\lambda e^{-\lambda t_0}\left(1-e^{-\lambda a t_W}\right) \\ &\quad - c_L\omega\lambda e^{-\lambda t_0}\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right).\end{aligned} \tag{13}$$

Let $\delta(t_0,t_W)$ denote the right-hand-side of Eq.(13). Then it is seen that

$$\delta(0,t_W) = k_0 + \omega\lambda\left[c_0 - c_W\left(1-e^{-\lambda a t_W}\right) - c_L\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right)\right] \tag{14}$$

and

$$\lim_{t_0\to+\infty}\delta(t_0,t_W) = k_0. \tag{15}$$

By taking the differentiation of Eq.(13) with respect to t_0, we get

$$\frac{\partial\delta(t_0,t_W)}{\partial t_0} = \omega\lambda^2 e^{-\lambda t_0}\left[-c_0 + c_W\left(1-e^{-\lambda a t_W}\right) + c_L\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right)\right]. \tag{16}$$

Letting $\gamma(t_0,t_W)$ denote the right-hand-side of Eq.(16), we obtain

$$\gamma(0,t_W) = \omega\lambda^2\left[-c_0 + c_W\left(1-e^{-\lambda a t_W}\right) + c_L\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right)\right]. \tag{17}$$

Further, let $A(t_W)$ denote the right-hand-side of Eq.(17). Then conditions $A(0) > 0$ and $A(t_L) > 0$ are reduced to

$$c_L\left(1-e^{-\lambda a t_L}\right) > c_0 \tag{18}$$

and

$$c_W\left(1-e^{-\lambda a t_L}\right) > c_0, \tag{19}$$

respectively. Since $c_L > c_W$ from assumption (A-I), we can show that $A(0) > A(t_L)$ and

$$\frac{\partial A(t_W)}{\partial t_W} = -(c_L - c_W)\omega\lambda^3 a e^{-\lambda a t_W} < 0. \tag{20}$$

Hence, we have $\gamma(0,t_W) > 0$. Since

$$\lim_{t_0\to+\infty}\gamma(t_0,t_W) = 0, \tag{21}$$

we can prove that $\partial\gamma(t_0,t_W)/\partial t_0 < 0$, say,

$$c_W\left(1-e^{-\lambda a t_W}\right) + c_L\left(e^{-\lambda a t_W} - e^{-\lambda a t_L}\right) > c_0. \tag{22}$$

The proof is completed. Q.E.D.

Furthermore, the following result provides the optimal planned maintenance limit which minimizes the total expected software cost.

Theorem 2: *When the software fault-detection time distribution follows the exponential distribution with mean $1/\lambda$, under the assumption (A-I), the optimal planned maintenance limit which minimizes the total expected software cost is given as follows:*

(1) If $k_W \geq (c_L - c_W)\omega\lambda a e^{-\lambda t_0}$, then the optimal policy is $t_W^* = 0$ with

$$C(t_0,t_W^*) = k_0 t_0 + c_0\omega\left(1-e^{-\lambda t_0}\right) + c_W\omega\left[e^{-\lambda t_0} - e^{-\lambda(t_0+at_L)}\right]. \tag{23}$$

(2) If $k_W < (c_L - c_W)\omega\lambda a e^{-\lambda t_0}$ *and* $k_W > (c_L - c_W)\omega\lambda a e^{-\lambda(t_0+at_L)}$, *then there exists a unique optimal planned maintenance limit* t_W^* $(0 < t_W < t_L)$ *which minimizes the total expected software cost with*

$$\begin{aligned} C(t_0, t_W^*) &= k_0 t_0 + c_0\omega\left(1 - e^{-\lambda t_0}\right) + k_W t_W^* + c_W\omega\left[e^{-\lambda t_0} - e^{-\lambda(t_0+at_W^*)}\right] \\ &\quad + c_W\omega\left[e^{-\lambda a(t_0+t_W^*)} - e^{-\lambda a(t_0+t_L)}\right]. \end{aligned} \tag{24}$$

(3) If $k_W \leq (c_L - c_W)\omega\lambda a e^{-\lambda(t_0+at_L)}$, *then we have* $t_W^* = t_L$ *with*

$$C(t_0, t_W^*) = k_0 t_0 + c_0\omega\left(1 - e^{-\lambda t_0}\right) + k_W t_L + c_W\omega\left[e^{-\lambda t_0} - e^{-\lambda(t_0+at_L)}\right]. \tag{25}$$

The proof of Theorem 2 is similar to that of Theorem 1, and is omitted for brevity.

Remark 3: The algorithm for finding the joint optimal policy combined by both the testing period and the planned maintenance limit is as follows:

Algorithm

Step 1 If there exists a solution (t_0, t_w) $(t_0 \geq 0,\ 0 \leq t_W \leq t_L)$ which satisfies the following first order condition of optimality, then go to Step 2, otherwise, go to Step 3.

$$\frac{\partial C(t_0, t_W)}{\partial t_0} = \frac{\partial C(t_0, t_W)}{\partial t_W} = 0. \tag{26}$$

Step 2 Define the Hessian matrix $H(t_0, t_W)$:

$$H(t_0, t_W) = \frac{\partial^2 C(t_0, t_W)}{\partial t_0^2}\frac{\partial^2 C(t_0, t_W)}{\partial t_W^2} - \left(\frac{\partial^2 C(t_0, t_W)}{\partial t_0 \partial t_W}\right)^2. \tag{27}$$

If $H(t_0, t_W) > 0$ and $\partial^2 C(t_0, t_W)/\partial t_0^2 > 0$, then the solution (t_0, t_W) found in Step 1 is regarded as the joint optimal policy, otherwise, go to Step 3.

Step 3 Find t_W^* $(0 \leq t_W^* \leq t_L)$ by solving $\partial C(0, t_W)/\partial t_W = 0$. Let $C_1(t_0^*, t_W^*) = C(0, t_W^*)$.

Step 4 Find t_0^* $(0 \leq t_0^* < \infty)$ by solving $\partial C(t_0, 0)/\partial t_0 = 0$. Let $C_2(t_0^*, t_W^*) = C(t_0^*, 0)$.

Step 5 Find t_0^* $(0 \leq t_0^* < \infty)$ by solving $\partial C(t_0, t_L)/\partial t_0 = 0$. Let $C_3(t_0^*, t_W^*) = C(t_0^*, t_L)$.

Step 6 The minimum total expected software cost is $C(t_0^*, t_W^*) = \min_{i=1,2,3} C_i(t_0^*, t_W^*)$ and the corresponding pair (t_0^*, t_W^*) is the joint optimal policy.

5 Numerical Examples

Based on 86 software fault data observed in the real software testing process [13], we calculate numerically the optimal testing period t_0^* and the optimal planned maintenance limit t_W^* which minimize the total expected software cost. Further, we compute the joint optimal policy (t_0^{**}, t_W^{**}) minimizing $C(t_0, t_W)$. For the software fault-detection time distribution, we apply three distributions; exponential [2], gamma of order 2 [14] and Rayleigh [15] distributions. The probability distribution functions for gamma of order 2 and Rayleigh are given by

$$F(t) = 1 - (1 + \lambda t)e^{-\lambda t} \tag{28}$$

and

$$F(t) = 1 - \exp\left\{-\frac{t^2}{2\theta^2}\right\}, \tag{29}$$

respectively. For the gamma distribution in Eq.(28), the total expected software cost in Eq.(6) becomes

$$\begin{aligned} C(t_0, t_W) &= k_0 t_0 + c_0\omega\left[1 - (1 + \lambda t_0)e^{-\lambda t_0}\right] + k_W t_W \\ &\quad + c_W\omega\left\{(1 + \lambda t_0)e^{-\lambda t_0} - [1 + \lambda(t_0 + at_W)]e^{-\lambda(t_0+at_W)}\right\} \end{aligned}$$

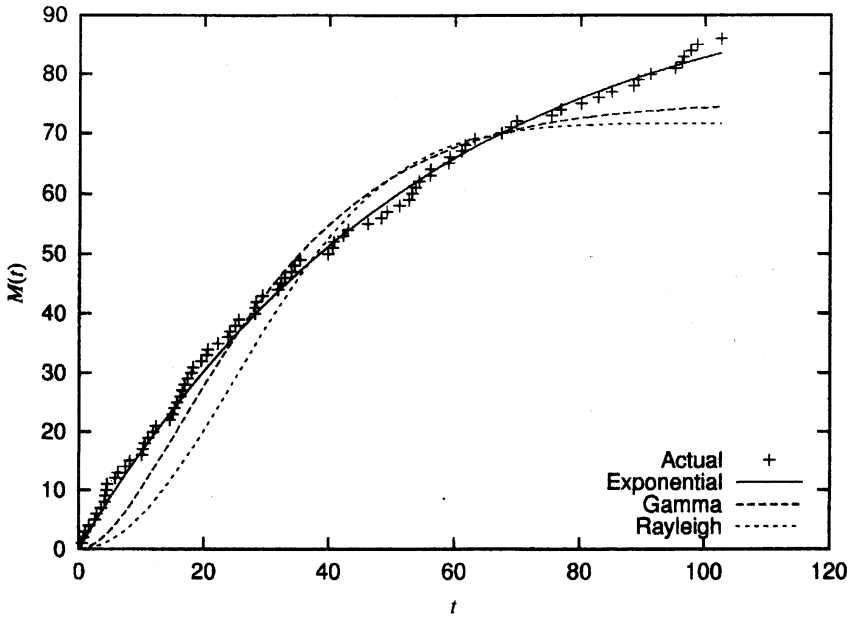

Figure 1: The actual software failure data and the behavior of estimated mean value functions.

$$+ c_L\omega\left\{[1+\lambda(t_0+at_W)]e^{-\lambda(t_0+at_W)} - [1+\lambda(t_0+at_L)]e^{-\lambda(t_0+at_L)}\right\}. \quad (30)$$

For the Rayleigh distribution, it becomes

$$\begin{aligned} C(t_0,t_W) &= k_0t_0 + c_0\omega\left[1-\exp\left\{-\frac{t_0^2}{2\theta^2}\right\}\right] + k_Wt_W \\ &\quad + c_W\omega\left[\exp\left\{-\frac{t_0^2}{2\theta^2}\right\} - \exp\left\{-\frac{(t_0+at_W)^2}{2\theta^2}\right\}\right] \\ &\quad + c_L\omega\left[\exp\left\{-\frac{(t_0+at_W)^2}{2\theta^2}\right\} - \exp\left\{-\frac{(t_0+at_L)^2}{2\theta^2}\right\}\right]. \end{aligned} \quad (31)$$

Suppose that the unknown parameters in the software reliability models are estimated by the method of maximum likelihood, when 70 fault data are obtained, namely $t = 67.374$. Then, we have the estimates $(\hat{\omega},\hat{\lambda}) = (98.5188,\ 1.84\text{e-}02)$ for the exponential model, $(\hat{\omega},\hat{\lambda}) = (75.1746,\ 6.46224\text{e-}02)$ for the gamma model and $(\hat{\omega},\hat{\theta}) = (71.6386,\ 2.45108\text{e+}01)$ for the Rayleigh model. Figure 1 shows the actual software fault data and the behavior of estimated mean value functions. For the other model parameters, we assume: $k_0 = 0.02$, $k_W = 0.01$, $c_0 = 1.0$, $c_W = 2.0$, $c_L = 20.0$ and $t_L = 1000$.

Table 1 presents the dependence of the environment factor a on the optimal testing period t_0^* when $t_W = 50$. As the environment factor monotonically increases, *i.e.*, the operational circumstance tends to be severe, it is observed that the optimal testing period t_0^* and its associated minimum total expected software cost $C(t_0^*, 50)$ decrease for both exponential and gamma models. For the Rayleigh model, it is observed that the optimal testing period is hardly influenced by varying environment factor. This is because the goodness-of-fit of the Rayleigh model is quite low.

Table 2 shows the dependence of the environment factor a on the optimal planned maintenance limit t_W^* in case of $t_0 = 70$. It is found that the optimal planned maintenance limit t_W^* and the corresponding minimum total expected software cost $C(70, t_W^*)$ decrease as the environment factor monotonically increases. This tendency can be explained as follows: The residual faults in software are detected and removed at the early stage in the operational phase as the operational environment becomes more severe. Then, the possibility that the software failure occurs in the latter stage of the operational phase becomes small. Hence, the implication in which the software developer keeps the maintenance team becomes smaller toward the end of the life cycle.

Table 3 examines the dependence of the environment factor a on the joint optimal policy (t_0^*, t_W^*) combined by testing period and planned maintenance limit. Figure 2 illustrates the behavior of the expected cost for the exponential model when $a = 2.00$. It is observed from Table 3 that the optimal

Table 1: Optimal testing period for varying environment factor.

	Exponential		Gamma		Rayleigh	
a	t_0^*	$C(t_0^*, 50)$	t_0^*	$C(t_0^*, 50)$	t_0^*	$C(t_0^*, 50)$
0.50	381.6	107.7	145.1	78.9	88.8	74.0
0.75	370.2	107.5	135.8	78.7	86.9	74.0
1.00	359.1	107.3	128.4	78.6	86.7	74.0
1.25	348.3	107.1	123.1	78.5	86.6	74.0
1.50	337.9	106.9	119.9	78.4	86.6	74.0
2.00	318.3	106.5	117.1	78.4	86.6	74.0
3.00	286.3	105.8	116.2	78.4	86.6	74.0

Table 2: Optimal planned maintenance limit for varying environment factor.

	Exponential		Gamma		Rayleigh	
a	t_W^*	$C(70, t_W^*)$	t_W^*	$C(70, t_W^*)$	t_W^*	$C(70, t_W^*)$
0.50	664.0	134.8	193.0	83.3	111.9	82.8
0.75	472.1	132.5	137.9	82.7	77.8	82.4
1.00	369.7	131.3	108.3	82.4	60.0	82.2
1.25	305.5	130.6	89.6	82.1	49.0	82.1
1.50	261.2	130.1	76.8	82.0	41.5	82.0
2.00	203.7	129.4	60.0	81.8	32.0	81.9
3.00	143.4	128.7	42.3	81.6	22.0	81.8

testing period t_0^* decreases as the environment factor monotonically increases, but, the monotonicity of the optimal planned maintenance limit t_W^* is not observed. It is also seen that the minimum total expected software cost $C(t_0^*, t_W^*)$ decreases as the environment factor monotonically increases.

Tables 4, 5 and 6 present the dependence of the software reliability model parameter λ or θ on the joint optimal policy (t_0^*, t_W^*) . Except for $\lambda = 0.0055$ in the gamma model, it is observed that the optimal testing time, optimal planned maintenance limit and its associated minimum total expected software cost decrease as the fault detection becomes easier.

6 Concluding Remarks

In this paper, we have assumed that the software developer was responsible to the maintenance service for all the software failures that occur during the software life cycle under the maintenance contract. In order to carry out the maintenance service in the operational phase, the software developer has to keep a software maintenance team. At the same time, the management cost in the operational phase has to be reduced as much as possible, but human resources should be utilized effectively. We have called the time length to complete the operational maintenance after the release the planned maintenance limit, and have controlled it in terms of cost-benefit analysis. We have developed the model which considers the difference in the software execution environment during testing and operational phases using the same method as the reliability assessment modeling in the operational phase proposed by Okamura *et al.* [12]. Based on NHPP, we have formulated the total expected software cost incurred to the software developer until the end of software life cycle. The optimal testing period (release time) and optimal planned maintenance limit which minimize the total expected software cost have been derived. Then, throughout the numerical examples, we have discussed the joint optimal policy combined by testing period and planned maintenance limit.

Table 3: Joint optimal policy for varying environment factor.

	Exponential			Gamma			Rayleigh		
a	t_0^{**}	t_W^{**}	$C(t_0^{**}, t_W^{**})$	t_0^{**}	t_W^{**}	$C(t_0^{**}, t_W^{**})$	t_0^{**}	t_W^{**}	$C(t_0^{**}, t_W^{**})$
0.50	405.0	0.0	107.7	167.4	0.0	78.9	106.3	0.0	73.9
0.75	304.6	159.2	107.3	135.6	50.4	78.7	94.5	18.4	73.8
1.00	282.6	157.1	106.8	128.5	49.8	78.6	91.7	18.3	73.8
1.25	272.7	143.3	106.5	125.2	45.5	78.5	90.4	16.7	73.7
1.50	267.0	129.8	106.2	123.4	41.2	78.4	89.6	15.1	73.7
2.00	260.6	108.4	105.9	121.3	34.3	78.3	88.8	12.6	73.7
3.00	254.9	81.5	105.5	119.4	25.8	78.2	88.0	9.4	73.6

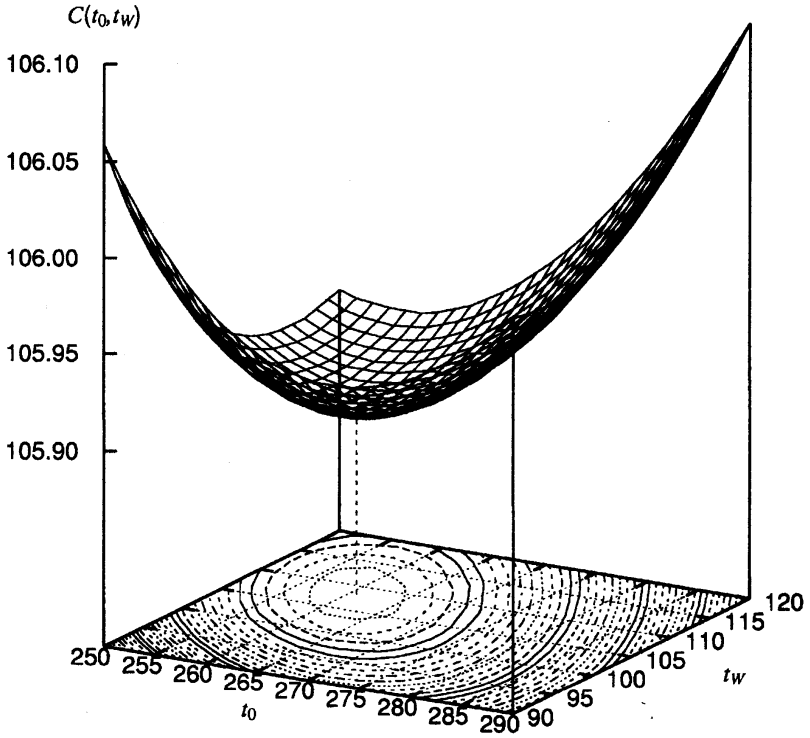

Figure 2: Behavior of the total expected software cost when $a = 2.00$.

Table 4: Joint optimal policy for varying exponential software reliability model parameter λ.

	Exponential		
λ	t_0^{**}	t_W^{**}	$C(t_0^{**}, t_W^{**})$
0.005	735.1	533.6	122.6
0.010	440.8	263.7	112.0
0.015	320.9	175.8	108.0
0.020	255.1	131.8	106.0
0.025	213.0	105.5	104.6
0.030	183.6	87.9	103.7
0.035	161.7	75.3	103.1

Table 5: Joint optimal policy for varying gamma software reliability model parameter λ.

	Gamma		
λ	t_0^{**}	t_W^{**}	$C(t_0^{**}, t_W^{**})$
0.0055	824.6	1000.0	106.1
0.0060	876.6	516.7	101.7
0.0065	826.6	474.0	100.0
0.0070	781.7	438.3	98.5
0.0075	741.5	407.9	97.2
0.0080	705.3	381.5	96.0
0.0085	672.6	358.4	94.9

Table 6: Joint optimal policy for varying Rayleigh software reliability model parameter θ.

	Rayleigh		
θ	t_0^{**}	t_W^{**}	$C(t_0^{**}, t_W^{**})$
22.5	83.5	15.2	73.6
23.0	85.2	15.6	73.6
23.5	87.0	16.0	73.7
24.0	88.7	16.3	73.7
24.5	90.4	16.7	73.7
25.0	92.1	17.1	73.8
25.5	93.7	17.4	73.8

References

[1] K. Okumoto and L. Goel, "Optimum release time for software systems based on reliability and cost criteria," J. Sys. Software, vol. 1, pp. 315–318, 1980.

[2] A.L. Goel and K. Okumoto, "Time-dependent error-detection rate model for software reliability and other performance measures," IEEE Trans. Reliab., vol. R-28, no. 3, pp. 206–211, 1979.

[3] H.S. Koch and P. Kubat, "Optimal release time of computer software," IEEE Trans. Software Eng., vol. SE-9, no. 3, pp. 323–327, 1983.

[4] Z. Jelinski, and P.B. Moranda, "Software reliability research," Statistical Computer Performance Evaluation, (W. Freiberger ed.), pp. 465–484, Academic Press, New York, 1972.

[5] D.S. Bai and W.Y. Yun, "Optimum number of errors corrected before releasing a software system," IEEE Trans. Reliab., vol. R-37, no. 1, pp. 41–44, 1988.

[6] W.Y. Yun and D.S. Bai, "Optimum software release policy with random life cycle," IEEE Trans. Reliab., vol. R-39, no. 2, pp. 167–170, 1990.

[7] T. Dohi, N. Kaio and S. Osaki, "Optimal software release policies with debugging time lag," Int. J. Reliab., Quality and Safety Eng., vol. 4, no. 3, pp. 241–255, 1997.

[8] M. Kimura, T. Toyota, and S. Yamada, "Economic analysis of software release problems with warranty cost and reliability requirement," Reliab. Eng. & Sys. Safe., vol. 66, no. 1, pp. 49–55, 1999.

[9] H. Pham and X. Zhang, "A software cost model with warranty and risk costs," IEEE Trans. Comput., vol. 48, no. 1, pp. 71–75, 1999.

[10] T. Dohi, H. Okamura, N. Kaio and S. Osaki, "The age-dependent optimal warranty policy and its application to software maintenance contract," Proc. 5th Int'l Conf. on Probab. Safe. Assess. and Mgmt. (S. Kondo and K. Furuta, eds.), vol. 4, pp. 2547–2552, University Academy Press Inc., 2000.